REMARKS

This is in response to the Office Action mailed on April 10, 2009 in which claims 1-17 were pending and rejected. With this Amendment, claims 1, 7, 8, 10, and 11 have been amended and claims 3-4 have been cancelled. Further, new claims 25-26 have been added. In view of the following, reconsideration and allowance are respectfully requested.

Claim Rejections – 35 U.S.C. § 103

Claims 1, 2, 5-7, 10-12, and 15-17 were rejected under 35 U.S.C. §103(a) as allegedly being unpatentable over Lisle et al. (U.S. Patent No. 4,843,389 – hereinafter "Lisle") in view of Katayama et al. (U.S. Patent No. 6,260,051 – hereinafter "Katayama") and further in view of Okada (U.S Patent No. 5,889,481 – hereinafter "Okada"). Claims 3, 4, 8, 9, 13, and 14 were rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over Lisle in view of Katayama and Okada and further in view of Edberg (U.S. Patent No. 5,873,111).

The present invention involves the collation of words or symbols. Collation means the sorting of text strings (which consist of symbols) according to an order of the symbols that is culturally correct to users of that language. Collation is used when users order linguistic data or perform a search for ordered linguistic data. For example, in the United States, words are collated such that those beginning with the letter "Q" are ordered after those beginning with the letter "P". In other languages, such as Chinese, linguistic symbols may be sorted by phonetic pronunciation or by the number of strokes in a symbols.

A compression is a special group of symbols that is treated as a single sort element. For example, in the Hungarian language, the letters "DZS" form a single sort element, as do the symbols "DZ". These are both "compressions" as used in the present specification. The compression "DZS" is treated as a single sort element and is arranged, in collation order, before the letter "E" in the Hungarian language, but after the compression "DZ".

The compression type of a given compression refers to the number of symbols that are grouped together as a single sort element. For instance, the compression type of the

compression "DZS" is 3 (or 3-to-1). The compression type of the compression "DZ" is 2 (or 2-to-1).

The highest compression type used in a given language varies based on the given language. For instance, some languages, such as Bengali or Tibetan, use compression types as high as 8-to-1. It is therefore very difficult to perform collation because, in order to do so, the collator must first determine whether some of the symbols in an input string to be collated are part of a compression, and should therefore be grouped together as a single sort element.

The present system deals with building tables that can be used in order to identify or sort compressions. One embodiment sets up symbol tables that include a list of code points that uniquely identify one of the symbols, and a sort weight for each of the identified symbols. The compression tables include a compression type that identifies a number of symbols in a given compression, and compressions of symbols of that compression type. Each compression is a grouping of two or more symbols that is treated as a single sort element for purposes of linguistic sorting such that an order in the linguistic sorting is determined based on the type of a given compression, a first of the two or more symbols in the given compression, and a predefined order of those symbols. Thus, for instance, the symbol tables have compressions that begin with the letter "D" sorted before those that begin with the letter "E". Each compression that beings with "D" is also sorted based on the compression type (i.e., based on the number of symbols contained in the compression).

This is simply neither taught nor suggested by any of the references cited by the Examiner.

As asserted by Applicant, and acknowledged by the Examiner in the present Office Action, the Lisle reference does not teach or suggest concepts related to compressions where a compression comprises a group of two or more symbols treated as a single unit for purposes of linguistic sorting. Lisle does <u>not</u> disclose compression tables or a compression type that identifies a number of symbols in a given compression. Even though Lisle does not teach or suggest compressions as claimed, the Office Action (see e.g., page 10) still alleges that Lisle

discloses compression tables, sorting compression table to identify a highest compression type, and storing a tag to indicate said highest compression type. This simply cannot be true. Nowhere does Lisle mention compressions or that symbols can be grouped and treated as a single sort element. The Office Action acknowledges this fact. Thus, Lisle cannot provide a table of compressions or compression types, at least based on the fact that Lisle does not disclose compressions.

In the "Response to Arguments" section beginning on page 2 of the Office Action, the Examiner states "[a] compression table is construed to be functionally equivalent and equally effective to that of a dictionary or list." Applicant strongly disagrees with this rationale as the Examiner disregards the claim language. Claim 1 specifically recites "each compression table pertaining to one of the supported languages and having a compression type identifying a number of symbols in a given compression in the compression table and containing compressions of symbols of that compression type, each compression being a grouping of two or more symbols treated as a single sort element for purposes of linguistic sorting" (emphasis added). A dictionary as disclosed by the cited references is not a compression table as claimed. For instance, as similarly described in Applicant's previous response filed on January 20, 2009, the Lisle reference involves dictionaries that are collated, with special characters first, followed by the alphabet, and then by the numbers. See column 15, lines 45-51. Lisle discusses two embodiments in which the dictionary may either have two byte or single byte entries. The entries are not compressions comprising groups of symbols treated as a single sort element. As can be seen from the disclosure of Lisle, the dictionaries disclosed therein are fundamentally different than a claimed "compression table."

Further, it is noted that columns 5 and 6 of Lisle discuss how many bit patterns can be used for control, and how many for word entries in a given dictionary. However, this is just discussing the size of the dictionary and not the individual entries in the dictionary. At this portion of Lisle, Lisle states that one bit is used to determine whether the dictionary is a one byte or two byte configuration. This has nothing to do with the number of symbols in a compression.

Instead, it is a discussion of the number of bytes which can be used to define each entry for a word or number or special character in the dictionary, regardless of how many symbols that word or number of special character is composed of. See column 15, lines 30-37, column 4, lines 48-55, column 5, line 21-column 6, line 50.

In sum, to the extent that Lisle uses the word "compression", Lisle is referring to compressing data in the dictionary, and not treating multi-letter elements (of multi-symbol elements) as a single sort element when performing collation. Lisle simply does not teach compression tables, or sorting compression tables, at all. Similarly, Lisle does not even teach or suggest a "compression type" which identifies a number of symbols in a given "compression" wherein "each compression is a grouping of two or more symbols treated as a single sort element for purposes of linguistic sorting".

These deficiencies are not cured by Katayama. In contrast to the allegations on pages 11 and 12 of the Office Action, Katayama does not teach or suggest a compression or a compression type as claimed. Instead, Katayama deals with information retrieval and matches a query character string against the letter strings in a document in the information retrieval process. See column 1, lines 7-18. In doing so, Katayama calculates the occurrence frequency of character strings in the documents and specifically refers to general characters, and special characters. General characters are characters such as letters, while special characters have no meaning but divide the text string of the document into two different strings each of which have meaning by themselves. For instance, a special character may be a space that divides two words in the textual string. See column 4, lines 49-52.

In order to determine a relationship between queries and documents, Katayama segments text in a document into character sequences and counts the frequency of occurrence of various character sequences in the document. Column 1, lines 23-53. The Katayama reference is primarily directed to determining how to accurately calculate the occurrence frequencies of various characters and character combinations.

One of the problems encountered by Katayama is that special characters, such as spaces for example, will have a very high number of occurrences in any document, regardless of the words that appear in that document. Therefore, if the occurrence frequency of spaces alone were considered in determining whether the query relates to the document, that will yield false results. See column 4, lines 33-58. Katayama therefore, does not calculate the occurrence frequency of just special characters, such as spaces. Instead, it calculates the occurrence frequency of those characters in combination with general characters so that the count is not skewed in favor of the special characters. Column 5, lines 10-57.

Specifically, in performing matches between the query and the document, Katayama does not look for matches for the special characters, when they occur as fore or aft characters, but only when they are sandwiched between general characters. The frequency of occurrence for such a three character chain does not include a frequency for the special character, but is only based on the fore and aft general characters in the three character chain. See column 68.

It will also be noted that Katayama unfortunately uses the word "collating" but it would appear that Kayayama does not use that word to mean "ordering". Instead, Katayama appears to be using the word collating to mean "matching". For instance, beginning at column 4, line 61, "an object of the present invention is to provide...a conventional character string collating apparatus...in which all pieces of character data of a text are recorded...and a character sting collating apparatus in which a retrieval character string is efficiently collated with a registration character string of a text...". Katayama also states, beginning at column 3, lines 46 "an occurrence frequency collating unit 14 for collating the occurrence frequency of the fore character in each occurrence frequency set...with that of the rear character in a particular occurrence frequency set of another particular two character chain type...". In these places, it would appear that Katayama is discussing how the occurrence frequency of various characters are matched against one another, and not ordered. In the first citation, it appears that Katayama is discussing how the retrieval character string [i.e., the query] is efficiently matched with

("collated with") a registration character string of a text (i.e., the text of a document). In the second citation, it appears that the occurrence frequency of a fore character in one character chain is matched against the occurrence frequency of the rear character in another character chain. Thus, it does not appear that Katayama is even directed to linguistic ordering as set out in the present claims.

In any case, Katayama is explicitly silent as to the concept of a compression which is a grouping of two or more symbols that are treated as a single sort element for purposes of linguistic sorting. The "two-character chain table" in the cited sections of Katayama (col. 130-131) relates to matching occurrence frequency of characters in the two-character chains; the two-character chains are not "compressions" that comprise groups of two or more symbols treated as a single sort element. Similarly, Katayama is silent as to even discussing the concept of a compression type which identifies a number of symbols in a given compression.

Further, in addition to failing to teach or suggest compressions or a compression table, Katayama does not teach or suggest that, for each code point in a symbol table, compression tables are sorted and a highest compression type is identified and a tag is stored in the symbol table for the code point to indicate said highest compression type. Katayama is void of any discussion of such features.

Moreover, as stated on page 13 of the Office Action, Lisle and Katayama do not teach or suggest linguistic sorting "determined based on a compression type of the given compression, a first of the two or more symbols in the given compression and a predefined order of symbols" as recited in claim 1. However, in contrast to the assertions in the Office Action these deficiencies are not cured by Okada.

The alleged "compressions" of Okada relates to "separat[ing] a data string inputted from source data of a Unicode in which different languages mixedly exist into a language string of each language and, after that, compress the data string every language string" (see Abstract). The data "compression" of Okada involves receiving and compressing a data string having a plurality of kinds of languages (col. 4, lns. 50-62). The data compression

apparatus determines the kinds of languages in the input, separates the data string into a language string of each language, and the each language string is compressed (col. 4, lns. 50-62). This data compression of Okada is used to reduce redundantly represented data and does not teach or suggest a "compression" comprising a grouping of two or more symbols treated as a single sort element for purposes of linguistic sorting as claimed. Moreover, nowhere does Okada teach or suggest that linguistic sorting is determined based on a compression type of the given compression, a first of the two or more symbols in the given compression, and a predefined order of symbols.

Further, as specifically recited in amended claim 1, "the tag for the code point is stored as a portion of the sort weight of the symbol identified by the code point, and wherein the sort weight of the symbol identified by the code point comprises a case weight value, and wherein the tag for the code point is stored as part of the case weight value for the code point." The cited references also do not teach or suggest, either separately or in combination, a tag for a code point stored as a portion of the sort weight or a sort weight of the symbol identified by the code point comprising a case weight value and where the tag for the code point is stored as part of the case weight value for the code point as claimed.

For at least the above reasons, Applicant respectfully submits that independent claim 1 is neither taught, suggested, nor rendered obvious by the cited references, and is in allowable form.

With regard to independent claim 6, Applicant respectfully submits that the cited references at least do not teach or suggest, either separately or in combination, providing a plurality of compression tables, each compression table having a compression type and containing compressions of symbols of that compression type, the compression type identifying a number of symbols in a compression, and each compression being a grouping of two or more symbols treated as a single sort element for purposes of linguistic sorting. Further, the cited references also do not teach or suggest, either separately or in combination, "for each code point in the symbol table, sorting the compression tables to order the compressions and to identify a

highest compression type for compressions, the order of the compressions being performed by ordering compressions based on a first of the two or more symbols and then ordering the compressions based on compression type, beginning with the symbol identified by said each code point." Further yet, the cited references also do not teach or suggest, either separately or in combination, "storing a tag in the symbol table for each code point to indicate said highest compression type for said each code point, wherein the tag for each code point is stored as a portion of the sort weight of the symbol identified by said each code point, and wherein the sort weight of the symbol identified by said each code point comprises a case weight value, and wherein the tag for said each code point is stored as part of the case weight value for said each code point."

For at least the above reasons, Applicant respectfully submits that independent claim 6 is neither taught, suggested, nor rendered obvious by the cited references, and is in allowable form.

With regard to independent claim 11, Applicant respectfully submits that the cited references at least do not teach or suggest, either separately or in combination, "receiving an input string containing a plurality of letters used in a given language" and "for a first letter in a combination of letters in the input string, referencing a symbol table to obtain a highest compression type for compressions beginning with said first letter, each compression being a grouping of two or more letters treated as a single sort element for purposes of linguistic sorting and the compression type identifying a number of letters in a given compression, the symbol table having a list of code points each uniquely identifying a letter and a sort weight for the letter identified by said each code point." For instance, Lisle simply fails to teach or suggest a multi-symbol element being treated as a single sort element. Instead, the alphabetization set out in Lisle treats each individual letter separately. It does not even discuss the idea of a compression, much less a compression type, or how to perform linguistic sorting based on the compression and compression types.

For at least the above reasons, Applicant respectfully submits that independent claim 11 is neither taught, suggested, nor rendered obvious by the cited references, and is in allowable form.

In conclusion, Applicant submits that independent claims 1, 6 and 11 are allowable. Applicant further submits that dependent claims 2, 5, 7-10, 12-17, and 25-26 which depend either directly or ultimately from the independent claims, are allowable as well. Reconsideration and allowance of the claims are respectfully requested.

The Director is authorized to charge any fee deficiency required by this paper or credit any overpayment to Deposit Account No. 23-1123.

Respectfully submitted,

MICROSOFT CORPORATION

By: /Christopher J. Volkmann/
Christopher J. Volkmann, Reg. No. 60,349
One Microsoft Way
Redmond, WA 98052-6399
Phone: (425) 707-9382

CJV:lah